INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO PIAUÍ
CAMPUS TERESINA CENTRAL
TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

RUAN MACEDO SANTOS

SMART STOCK: desenvolvimento de clientes mobile e web para controle de inventário utilizando RFID

TERESINA, PIAUÍ
2025

RUAN MACEDO SANTOS

SMART STOCK: desenvolvimento de clientes mobile e web para controle de inventário utilizando RFID

Trabalho de Conclusão de Curso (Relatório Técnico de Software) apresentado como exigência parcial para obtenção do diploma do Curso de Análise e Desenvolvimento de Sistemas do Instituto Federal de Educação, Ciência e Tecnologia do Piauí, Campus Teresina Central.

Orientador: Prof. M^{e}. Fernando Castelo Branco Gonçalves Santana

TERESINA, PIAUÍ
2025

Dados Internacionais de Catalogação na Publicação (CIP) de acordo com ISBD

Santos, Ruan Macedo

S237s Smart stock : desenvolvimento de clientes mobile e web para controle de inventário utilizado rfid / Ruan Macedo Santos. - 2025.

33 f.: il. color.

Trabalho de Conclusão de Curso (Tecnologia em Análise e Desenvolvimento de Sistemas) - Instituto Federal de Educação, Ciência e Tecnologia do Piauí, Campus Teresina Central, 2025.

Orientador : Prof Me. Fernando Castelo Branco Gonçalves Santana.

1. Embarcatech. 2. Sistema web. 3. Aplicativo móvel. 4. RFID. I.Título.

CDD - 004.21

Elaborado por Tanize Maria Sales CRB 3/1024

RUAN MACEDO SANTOS

SMART STOCK: desenvolvimento de clientes mobile e web para controle de inventário utilizando RFID

Trabalho de Conclusão de Curso (Relatório Técnico de Software) apresentado como exigência parcial para obtenção do diploma do Curso de Análise e Desenvolvimento de Sistemas do Instituto Federal de Educação, Ciência e Tecnologia do Piauí, Campus Teresina Central.

Aprovada em 18 de Dezembro de 2025.

BANCA EXAMINADORA:

Prof. Mᵉ. Fernando Castelo Branco Gonçalves Santana
(Orientador)
Instituto Federal do Piauí (IFPI)

Profª. Drª. Aline Montenegro Leal Silva
Instituto Federal do Piauí (IFPI)

Profª. Esp. Sandra Elisa Veloso Aguiar
Instituto Federal do Piauí (IFPI)

Dedico este trabalho à minha mãe Raimunda

AGRADECIMENTOS

Agradeço primeiramente a Deus por ter chegado até aqui, e aos meus pais pelo apoio incondicional e todos os sacrifícios que fizeram por mim. À minha irmã Ruana, à minha tia Janete, e aos meus tios Adriana e Gaspar deixo minha profunda gratidão por terem me ajudado a manter-me no curso.

Registro minha sincera gratidão ao meu orientador, Professor Fernando Santana, por possibilitar a realização deste trabalho e por todas as orientações ao longo do processo.

Agradeço ao Instituto Federal de Educação, Ciência e Tecnologia do Piauí (IFPI) pelo apoio institucional e à Associação para Promoção da Excelência do Software Brasileiro (Softex) pelo fomento e concessão da bolsa de extensão. O apoio financeiro foi viabilizado por meio do programa Residência em TIC 37 – Sistemas Embarcados, executado com a interveniência da Fundação de Amparo à Pesquisa, Inovação, Ensino e Extensão do IFPI (FAIFPI), sendo fundamental para o desenvolvimento desta pesquisa.

Deixo um agradecimento especial à Bianca, pela parceria constante durante todo o curso de ADS, e ao Meireles, Lívia, Ryan, Hermínio e Patrocínio por também terem me acolhido nesse percurso. Obrigado por me inspirarem a concluir esse trabalho. Agradeço ainda a Halyson, por ter sido um veterano tão solícito, e a Fernanda Siqueira e Venceslau Felipe por terem contribuído para meu amadurecimento ao longo dessa trajetória.

Essa jornada não foi feita sozinha, e carrega o mérito de todas as pessoas que me acompanharam até aqui. Por isso, agradeço a todos os meus colegas de curso, aos demais professores, às tias do refeitório e aos servidores do campus, pelo apoio que me deram direta ou indiretamente.

RESUMO

O controle da quantidade de mercadorias disponíveis representa um desafio para as empresas do setor terciário, pois os sistemas tradicionalmente fornecem essa variável a partir do registro de mudanças no estoque, o que limita sua veracidade à execução correta desses registros, que muitas vezes é feita de modo manual. Nessa lacuna, surge a realização de inventários, momentos em que é realizada a contagem física das mercadorias para garantir que o saldo real corresponda ao saldo registrado no sistema. Entre as diversas soluções que automatizam o processo de inventário, uma das mais populares é o uso de leitores RFID portáteis. No entanto, o alto custo de aquisição desses leitores limita sua adesão por pequenas e médias empresas. No contexto do programa EmbarcaTech, o projeto SmartStock propõe um leitor RFID portátil de baixo custo. O sistema completo é dividido entre firmware, mobile, backend e frontend, sendo que este trabalho apresenta e discute o desenvolvimento dos clientes web e móvel do sistema. Essas interfaces foram construídas utilizando respectivamente os frameworks Next.js e Flutter, garantindo aplicações modernas e confiáveis. Testes em ambiente controlado validaram a integração dos clientes com os demais componentes, e o resultado é um leitor ágil e de modelo replicável.

Palavras-chaves: embarcatech; sistema web; aplicativo móvel; rfid.

ABSTRACT

Managing the quantity of available goods poses a challenge for companies in the tertiary sector, as systems traditionally provide this variable based on records of inventory changes. The accuracy of such data depends on the correct execution of these records, which are often performed manually. To address this gap, inventory audits are conducted through physical counts to ensure the actual stock corresponds to the balance recorded in the system. Among the various solutions that automate the inventory process, the use of portable Radio-Frequency Identification (RFID) readers is one of the most popular. However, the high acquisition cost of these devices limits their adoption by small and medium-sized enterprises. Within the scope of the EmbarcaTech program, the SmartStock project proposes a low-cost portable RFID reader. The complete system comprises firmware, mobile, backend, and frontend components; this work specifically presents and discusses the development of the system's web and mobile clients. These interfaces were developed using the Next.js and Flutter frameworks, respectively, ensuring modern and reliable applications. Tests in a controlled environment validated the integration of these clients with the remaining components, resulting in an agile and replicable reader model.

Keywords: embarcatech; web system; mobile app; rfid.

LISTA DE ILUSTRAÇÕES

Figura 1 – Diagrama de casos de uso do Operador 16
Figura 2 – Diagrama de casos de uso do Administrador 16
Figura 3 – Diagrama da arquitetura 18
Figura 4 – Tela de boas-vindas 19
Figura 5 – Tela de login 19
Figura 6 – Tela principal 20
Figura 7 – Tela de leitores 20
Figura 8 – Modal de leitura de produto 21
Figura 9 – Modal de leitura limpa 21
Figura 10 – Tela de *setup* de gravação 22
Figura 11 – Tela de histórico de gravação 22
Figura 12 – Tela de gravação de produto 22
Figura 13 – Tela de gravação de limpeza 22
Figura 14 – Tela de *setup* de inventário ao criar um novo 23
Figura 15 – Tela de *setup* de inventário ao adentrar um existente 23
Figura 16 – Tela de realização de inventário 24
Figura 17 – Tela de pausa de inventário 24
Figura 18 – Tela de sincronização de inventário 24
Figura 19 – Tela de *setup* de inventário offline 24
Figura 20 – Componente deslizável da tela principal 25
Figura 21 – Modal de atualização do leitor 25
Figura 22 – Página de Login 26
Figura 23 – Página de Inventários 27
Figura 24 – Relatório de inventário 27
Figura 25 – Página de Usuários 28

LISTA DE ABREVIATURAS E SIGLAS

API	*Application Programming Interface*
APP	*Application*
HTTPS	*Hypertext Transfer Protocol Secure*
JS	*JavaScript*
JSON	*JavaScript Object Notation*
PIB	*Produto Interno Bruto*
REST	*Representational State Transfer*
RFID	*Radio-Frequency IDentification*
UML	*Unified Modeling Language*

SUMÁRIO

1 INTRODUÇÃO 11
1.1 OBJETIVOS 12
1.1.1 Objetivo Geral 12
1.1.2 Objetivos Específicos 12

2 TECNOLOGIAS ENVOLVIDAS 13
2.1 TECNOLOGIAS DO APLICATIVO MÓVEL 13
2.2 TECNOLOGIAS DO CLIENTE WEB 13
2.3 TECNOLOGIAS DE DESIGN 13

3 MODELAGEM DO PROJETO 14
3.1 LEVANTAMENTO DE REQUISITOS 14
3.1.1 Requisitos Funcionais 14
3.1.2 Requisitos Não Funcionais 15
3.2 DIAGRAMA DE CASOS DE USO 16
3.3 ARQUITETURA DO SISTEMA 18

4 APRESENTAÇÃO DO SISTEMA 19
4.1 CLIENTE MÓVEL 19
4.2 CLIENTE WEB 25

5 CONSIDERAÇÕES FINAIS 29
5.1 IMPLANTAÇÃO 29
5.2 RESULTADOS DO DESENVOLVIMENTO 29
5.3 TRABALHOS FUTUROS 29

REFERÊNCIAS 30

APÊNDICE A – PROTÓTIPO DO LEITOR 31

1 INTRODUÇÃO

A economia pode ser dividida em três setores principais: primário, secundário e terciário (ALMEIDA; SILVA; ANGELO, 2013). O setor terciário agrupa todas as atividades formais e informais de comércio e prestação de serviços. Em 2023, ele foi responsável por cerca de 4,27% do PIB brasileiro, com um aumento de 2,77% na estimativa realizada para sua participação naquele ano, o que demonstra sua importância e expansão no país (Fundação Instituto de Pesquisas Econômicas, 2023).

Nesse cenário, em empresas que comercializam bens materiais, para uma gestão estratégica e eficiente, é essencial controlar a quantidade de produtos disponíveis para venda. Apesar de essa informação ser tipicamente obtida a partir da diferença entre a quantidade de entrada e saída de itens, erros durante o registro dessas operações podem resultar em dados imprecisos, problema que se torna ainda mais grave quando esses registros são realizados manualmente. Sob essa ótica, muitas empresas complementam seus sistemas de controle de estoque com a realização de inventários. O inventário refere-se à contagem física das mercadorias para garantir que o saldo real corresponda ao saldo registrado no sistema.

Entretanto, quando o inventário é feito de forma totalmente manual, ele se torna propenso a demora e a erros, e é por isso que existem diversas soluções no mercado para automatizar esse processo. Atualmente, uma das mais adotadas é o uso de leitores portáteis com a tecnologia de identificação por radiofrequência (RFID), que permitem a contagem rápida, simultânea e sem contato físico direto de diversas etiquetas RFID, tornando o inventário semiautomático. No entanto, um dos principais problemas na adesão a essa solução é seu custo, visto que o alto custo de aquisição dos leitores convencionais limita a sua adoção por pequenas e médias empresas.

É nesse contexto então, que dentro da 1ª edição do programa de Residência em Sistemas Embarcados EmbarcaTech (Softex, 2025), o projeto SmartStock foi concebido como um leitor portátil RFID de baixo custo para auxiliar no processo de inventários. O ecossistema desse projeto é composto por quatro componentes: o firmware embarcado no leitor, um servidor backend para gerenciamento de lógica, um aplicativo móvel para interação entre usuário e leitor, e um cliente Web para gestão dos dados gerados pelo sistema.

O foco do presente trabalho são os clientes Web e móvel desenvolvidos. A metodologia envolveu o planejamento da solução utilizando ferramentas de engenharia de software e desenvolvimento com métodos ágeis, empregando tecnologias modernas de código aberto.

1.1 OBJETIVOS

Abaixo são listados os objetivos do presente trabalho:

1.1.1 Objetivo Geral

Desenvolver os clientes frontend e mobile do sistema de contagem de inventário SmartStock. O principal propósito do aplicativo móvel é oferecer uma interface entre o operador e o sistema embarcado, integrando a leitura e gravação RFID do sensor com os dados de produtos oriundos da API. A responsabilidade do sistema frontend é fornecer um painel de gestão pelo qual é possível acompanhar o histórico de inventários e controlar o acesso dos usuários ao leitor.

1.1.2 Objetivos Específicos

- Projetar as interfaces de usuário (UI) e a experiência do usuário (UX) focadas na agilidade operacional;
- Desenvolver o aplicativo móvel, implementando a comunicação via Bluetooth com o leitor RFID e a sincronização em tempo real com a API;
- Construir o cliente Web, integrando funcionalidades de gerência e controle de acesso dos usuários do leitor RFID, e de gestão dos inventários realizados.

2 TECNOLOGIAS ENVOLVIDAS

Este capítulo apresenta as tecnologias e ferramentas selecionadas para a fundamentação e execução do projeto, justificando a escolha de cada componente.

2.1 TECNOLOGIAS DO APLICATIVO MÓVEL

No desenvolvimento do aplicativo móvel, foi escolhido o *framework* Flutter, baseado na linguagem de programação Dart. Proposta inicialmente como substituta de JavaScript, Dart é uma linguagem orientada a objetos utilizada para construir aplicações multiplataforma. Flutter e Dart são uma excelente combinação para construir aplicações de alta qualidade, performance e escalabilidade (MARIMUTHU *et al.*, 2023).

2.2 TECNOLOGIAS DO CLIENTE WEB

A linguagem JavaScript foi empregada na codificação do cliente Web. Seguindo a padronização ECMAScript e baseada em objetos, JavaScript é conhecida por ser essencialmente incorporada em todos os navegadores de internet, apesar de vir sendo amplamente utilizada em servidores (ROKICKI; MAURICE; LAPERDRIX, 2021). E para melhorar a sua experiência de desenvolvimento, foi utilizado TypeScript, uma extensão sintática que permite escrever código com tipagem estática e depois compilá-lo em JavaScript (YEE; GUHA, 2023).

Para agilizar o processo de desenvolvimento, adotou-se React, uma biblioteca JavaScript para construção de interfaces de usuário focada em reutilização. E para complementá-la, utilizou-se Next.js, um *meta-framework* React que possibilita a renderização de aplicações do lado do servidor, garantindo sua performance (JARTARGHAR *et al.*, 2022). Por fim, visando facilitar o processo de implantação do cliente em ambiente de produção e integração com o restante do sistema, a aplicação Web e suas dependências foram conteinerizadas por meio do Docker (Docker, Inc., 2025).

2.3 TECNOLOGIAS DE DESIGN

Na prototipagem das telas do projeto, foi usado o Figma, uma plataforma de design de interfaces, prototipagem e gerenciamento de sistemas de design, popular entre os profissionais de design por sua versatilidade (IBRAHIM *et al.*, 2023). Figma ajuda a garantir a consistência visual no design permitindo a criação e reutilização de recursos como componentes e variáveis (Figma, Inc., 2025). A iconografia do projeto é composta principalmente pela biblioteca Lucide Icons, que possui licença permissiva ISC e conta com um acervo limpo e customizável (Lucide Contributors, 2025).

3 MODELAGEM DO PROJETO

Nesta seção serão discutidos aspectos relacionados ao projeto do software, apresentando alguns artefatos UML.

3.1 LEVANTAMENTO DE REQUISITOS

O levantamento de requisitos foi feito baseado em discussão com outros desenvolvedores, da qual os requisitos mais relevantes seguem pontuados abaixo. Os primeiros requisitos de cada tópico são os do aplicativo móvel, seguidos dos requisitos do frontend.

3.1.1 Requisitos Funcionais

- **[RFAPP01]** O aplicativo deve permitir que os operadores se autentiquem para poderem acessar a aplicação.
- **[RFAPP02]** O aplicativo deve gerenciar por conta própria o ciclo de conexão Bluetooth com o leitor.
- **[RFAPP03]** O aplicativo deve listar os leitores próximos e permitir que o operador troque o dispositivo conectado.
- **[RFAPP04]** O aplicativo deve buscar a lista de produtos do servidor e disponibilizá-la para os seus módulos.
- **[RFAPP05]** O aplicativo deve armazenar em cache a lista de produtos e permitir sua revalidação.
- **[RFAPP06]** O aplicativo deve permitir o registro das leituras feitas durante um inventário, comunicando-as ao servidor.
- **[RFAPP07]** O aplicativo deve permitir que o operador realize o registro de um inventário localmente mesmo que não haja conexão com a internet.
- **[RFAPP08]** O aplicativo deve permitir que o operador verifique rapidamente o conteúdo de uma etiqueta RFID fora de um inventário.
- **[RFAPP09]** O aplicativo deve permitir que o operador grave o código de um produto em uma etiqueta.
- **[RFAPP10]** O aplicativo deve permitir que o operador grave uma etiqueta como "limpa", e lidar com a detecção de etiquetas nesse estado durante seus processos de leitura.

- **[RFAPP11]** O aplicativo deve registrar e exibir localmente o histórico de codificações feitas, tanto de gravações, quanto de limpezas.
- **[RFAPP12]** O aplicativo deve permitir a atualização *over-the-air* do firmware do leitor.
- **[RFWEB01]** O frontend deve permitir que os administradores se autentiquem para poderem acessar a aplicação Web.
- **[RFWEB02]** O frontend deve listar todos os inventários registrados, além dos detalhes de cada um.
- **[RFWEB03]** O frontend deve permitir a conclusão e o cancelamento de um inventário em aberto.
- **[RFWEB04]** O frontend deve listar todos os usuários do sistema.
- **[RFWEB05]** O frontend deve permitir que os administradores criem outros usuários, operadores ou administradores, e desativem seu acesso se necessário.
- **[RFWEB06]** O frontend deve possibilitar a geração de relatórios dos inventários.

3.1.2 Requisitos Não Funcionais

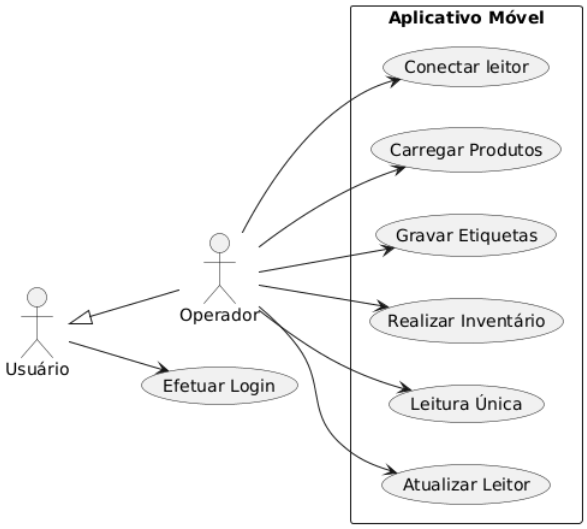
- **[RNFAPP01]** A comunicação via Bluetooth deve ser implementada utilizando Bluetooth Low Energy.
- **[RNFAPP02]** O aplicativo móvel deve ser compatível com as plataformas Android e iOS.
- **[RNFAPP03]** A interface de usuário deve ser acessível, e considerar o uso com uma única mão devido à necessidade de segurar o leitor portátil.
- **[RNFAPP04]** O aplicativo deve ser escalável e projetado para suportar futuros módulos com outras funcionalidades.
- **[RNFWEB01]** A interface gráfica deve ser funcional, fácil de utilizar, responsiva e operar suavemente em dispositivos utilizados pelos usuários.
- **[RNFWEB02]** A aplicação deve ser leve, para permitir o uso em eventuais condições adversas de conexão com a Internet e de capacidade de hardware dos usuários.
- **[RNFWEB03]** Garantir compatibilidade com os navegadores de internet modernos.

- **[RNFWEB04]** O código-fonte produzido durante o desenvolvimento do sistema deve seguir boas práticas de programação e estar bem documentado.

3.2 DIAGRAMA DE CASOS DE USO

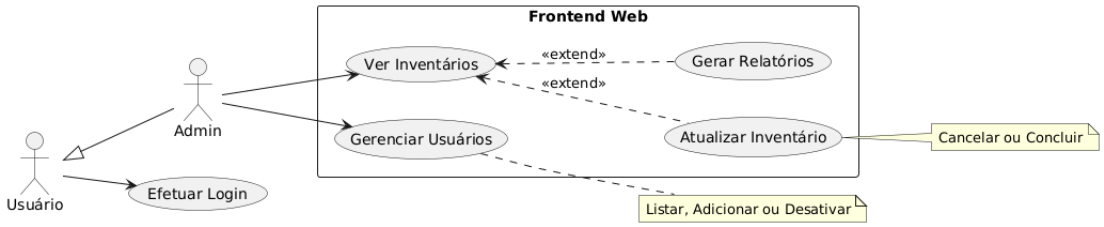
Os casos de uso foram montados com base no levantamento de requisitos apresentado. Os dois atores apontados na modelagem foram o Operador e Administrador, sendo que em ambos os casos, cada ator interage unicamente com um dos clientes do sistema. Os casos de uso do Operador são apresentados na Figura 1, enquanto os casos do Administrador podem ser vistos na Figura 2.

Figura 1 – Diagrama de casos de uso do Operador



Fonte: Criado pelo autor

Figura 2 – Diagrama de casos de uso do Administrador



Fonte: Criado pelo autor

Abaixo segue o detalhamento dos casos de uso:

1. **Efetuar login**: Interação entre o usuário e o sistema. Usuário se autentica no sistema, o que permite acesso às demais funcionalidades. As credenciais de entrada utilizadas são nome de usuário e senha. O cliente deve armazenar de forma segura o *token* de resposta devolvido pela API, já que ele será utilizado como identificador para o acesso daquele usuário aos demais recursos do servidor.

2. **Conectar leitor**: Interação entre o operador e o aplicativo. O operador abre o aplicativo, e aguarda que se conecte via Bluetooth ao leitor disponível mais próximo. O operador também tem acesso à lista de todos os leitores disponíveis nas proximidades do aparelho móvel, e pode trocar o dispositivo conectado por qualquer outro da lista. O acesso aos demais módulos do sistema só fica disponível após o emparelhamento bem-sucedido de um leitor.

3. **Carregar produtos**: Interação entre o operador e o aplicativo. O app carrega a lista de produtos do backend em memória não-volátil utilizando arquivos *JSON*, e permite que o usuário refaça essa requisição caso ache necessário. O operador vê por meio de um indicador quando a requisição foi feita há mais tempo do que o período onde esse dado é considerado "fresco"pelo sistema. Com exceção do *módulo de atualização*, não é possível acessar os demais componentes da aplicação enquanto a lista não está disponível.

4. **Gravar etiquetas**: Interação entre o operador e o aplicativo. Operador seleciona o código de um produto ou a opção de limpeza e sobrescreve o conteúdo gravado em uma etiqueta. Todas as operações são registradas no aparelho, salvando código do produto (ou de limpeza) e marca temporal, e o operador pode consultar o histórico completo.

5. **Realizar inventário**: Interação entre o operador e o aplicativo. Operador inicia um novo inventário diário ou adentra um que já tenha sido aberto por outro operador. Enquanto estiver na tela de Inventário, todas as leituras de etiquetas realizadas pelo leitor serão contabilizadas na sessão de inventário. O operador pode pausar o inventário, e durante esse estado, o aplicativo ignora novas leituras comunicadas pelo leitor. Devido ao seu alto volume, as leituras são primeiro salvas localmente, e de forma periódica são enviadas ao servidor. O operador consegue ver o último horário em que essa atualização ocorreu e sincronizar manualmente se necessário.

6. **Leitura única**: Interação entre o operador e o aplicativo. Operador na tela principal do aplicativo aponta o leitor para uma etiqueta, e após apertar o gatilho, consegue ver seu conteúdo em um modal. Etiquetas "limpas"também são exibidas nessa visualização.

7. **Atualizar leitor**: Interação entre o operador e o aplicativo. Operador por meio do aplicativo verifica no servidor a disponibilidade de atualização do firmware do leitor, e em caso positivo, pode executá-la. O aplicativo deve realizar a atualização via Bluetooth utilizando a metodologia *over-the-air*.

8. **Ver inventários**: Interação entre o administrador e o frontend. Administrador acessa uma lista de todos os inventários registrados no sistema. Os detalhes de cada inventário também são exibidos a partir dessa lista, sendo possível visualizar todas as leituras registradas em cada um.

9. **Atualizar inventário**: Interação entre o administrador e o frontend. Administrador pode modificar o estado de um inventário em aberto, podendo tanto cancelá-lo, quanto concluí-lo. Caso o administrador não conclua manualmente um inventário, o sistema automaticamente o marca como concluído ao final do dia.

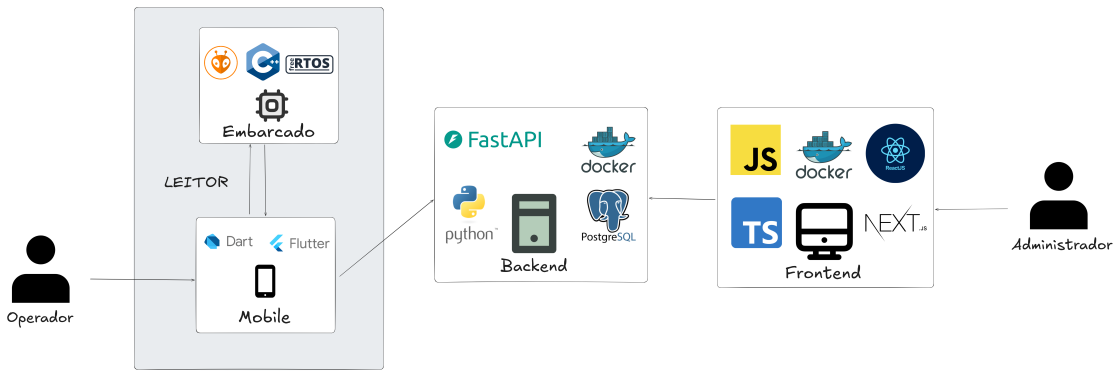
10. **Gerar relatórios**: Interação entre o administrador e o frontend. Administrador gera um arquivo de relatório contendo todas as informações de um relatório a partir da visualização dos detalhes de um inventário.

11. **Gerenciar usuários**: Interação entre o administrador e o frontend. Administrador acessa uma lista de todos os usuários cadastrados no sistema. O ator pode então adicionar novos usuários ao sistema, tanto com perfis de Administrador quanto de Operador, e também desativar o acesso de um usuário.

3.3 ARQUITETURA DO SISTEMA

A estrutura e organização dos componentes do sistema em um alto nível pode ser vista na Figura 3, bem como as principais tecnologias utilizadas.

Figura 3 – Diagrama da arquitetura



Fonte: Criado pelo autor

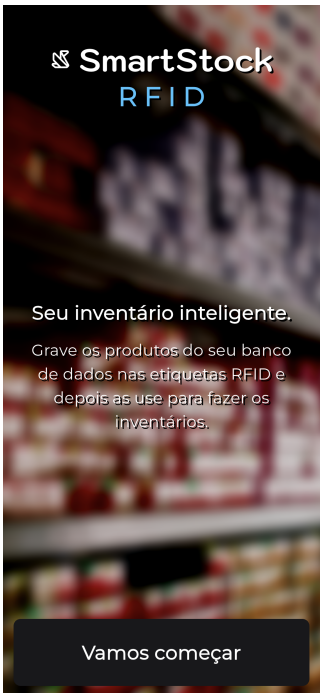
4 APRESENTAÇÃO DO SISTEMA

Nesta seção, será discutido o fluxo de utilização da aplicação a partir da apresentação das principais telas desenvolvidas nos clientes. As interfaces apresentam paletas de cores semelhantes, baseadas em tons neutros, sendo a cor primária do aplicativo mobile o tom *#18181B* e, no frontend Web, o tom *#101828*.

4.1 CLIENTE MÓVEL

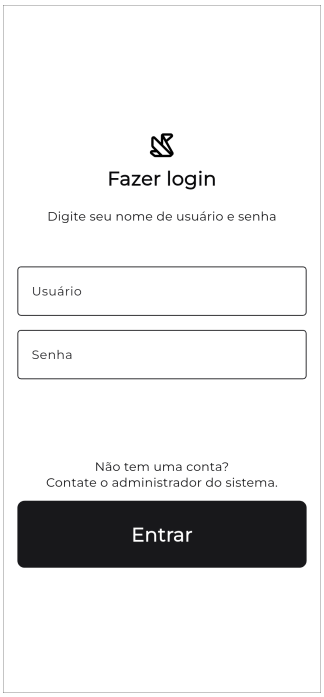
O fluxo do aplicativo móvel começa na tela de boas-vindas (Figura 4), que só é acessível na primeira vez em que o usuário abre o aplicativo. Ao prosseguir, o usuário é direcionado para a tela de login (Figura 5), onde precisa se autenticar para poder acessar o restante da aplicação. Quando a sessão do usuário expira, ou seu *refresh token* é invalidado no servidor backend, o roteador redireciona novamente o aplicativo para essa tela.

Figura 4 – Tela de boas-vindas



Fonte: Criado pelo autor

Figura 5 – Tela de login



Fonte: Criado pelo autor

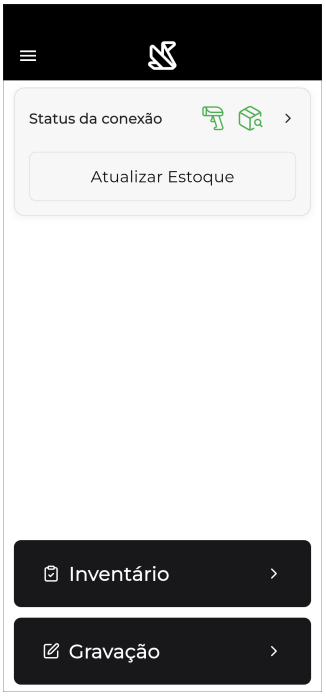
Após o processo de autenticação ser bem-sucedido, o usuário segue para a tela principal (Figura 6), a partir da qual ele acessa todas as demais. Nesse momento, em plano de fundo o aplicativo dispara uma consulta à lista de produtos do servidor backend e tenta se conectar com o primeiro leitor disponível que encontrar. O componente de

status no topo dessa tela possui dois ícones, um no formato de um leitor, que representa a conexão Bluetooth, e o outro em formato de caixa, representando a consulta de produtos. As cores dos símbolos refletem diretamente o estado da operação, sendo a cor verde para sucesso, azul para pendente e vermelho para erro.

Além disso, no caso dos produtos, ainda é utilizada a cor amarela para mostrar ao operador quando os dados da lista de produtos salvos em memória não são mais considerados "frescos"pelo aplicativo. O critério para considerar que os dados ainda estão válidos é configurável e leva em conta o intervalo de tempo entre o horário de atualização e o tempo atual. O usuário pode então utilizar esse *feedback* para decidir quando atualizar os dados, já que após a primeira consulta e salvamento em memória, o cache da lista de produtos só será atualizado se o usuário manualmente o fizer apertando o botão "Atualizar estoque".

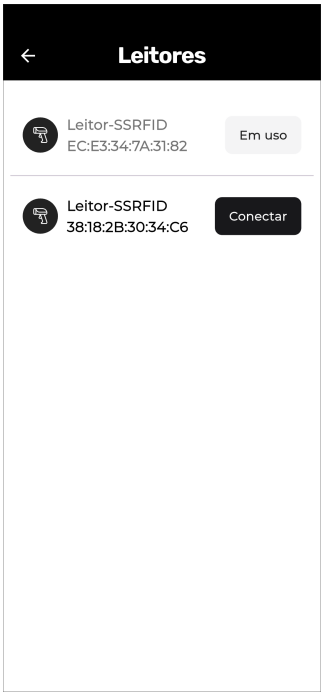
A seta no final do componente de status é utilizada para indicar ao usuário a possibilidade de navegação. Ao clicar em uma área do componente de status que não seja o botão de atualização de estoque, o aplicativo navegará para a tela de leitores (Figura 7). Nesta seção, é possível visualizar os leitores disponíveis e trocar o que está sendo utilizado atualmente pelo dispositivo móvel.

Figura 6 – Tela principal



Fonte: Criado pelo autor

Figura 7 – Tela de leitores



Fonte: Criado pelo autor

O próximo fluxo que pode ser realizado na tela principal é o de leitura única, exibido para o operador como ’Leitura rápida’. Ao apontar o leitor portátil para uma etiqueta RFID e pressionar o gatilho estando nessa seção, será exibido um modal com

o conteúdo da etiqueta (Figura 8). Conforme a Figura 9, caso a etiqueta não esteja cadastrada com os dados de um produto, também é fornecido um *feedback* adequado ao usuário.

Figura 8 – Modal de leitura de produto

Fonte: Criado pelo autor

Figura 9 – Modal de leitura limpa

Fonte: Criado pelo autor

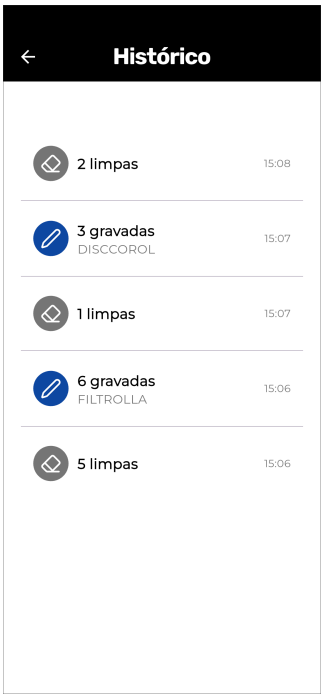
A partir da tela principal, ao clicar no botão de navegação "Gravação", o usuário será redirecionado para a tela de preparação de gravação (Figura 10). Nesse momento, é possível escolher se o tipo de gravação realizada vai ser de um produto, ou de restauração (ou limpeza). Optando pelo modo de produto, o usuário ainda precisa escolher que produto deseja gravar a partir de um componente de pesquisa encontrado abaixo da opção "Produto". No canto superior dessa tela, é possível visualizar um botão com ícone de relógio, pelo qual é possível acessar a tela de histórico de gravações, e visualizar todas as operações de gravação registradas localmente naquele dispositivo móvel, como mostrado na Figura 11.

Após apertar o botão primário da tela de preparação, o usuário é direcionado para a tela de gravação, onde efetivamente o leitor realizará a gravação das etiquetas. A aparência dessa tela difere dependendo do modo de gravação escolhido, sendo o modo de Produto apresentado na Figura 12, e o modo de Limpeza na Figura 13.

Figura 10 – Tela de *setup* de gravação

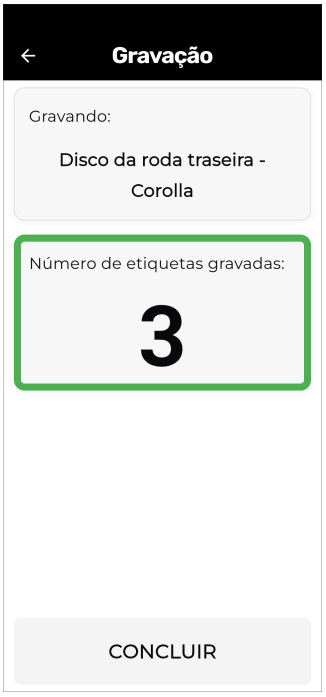
Fonte: Criado pelo autor

Figura 11 – Tela de histórico de gravação



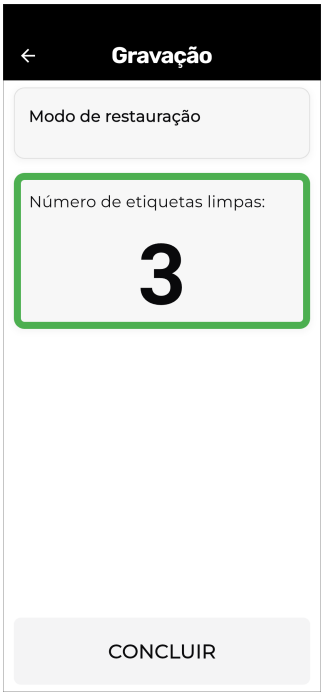
Fonte: Criado pelo autor

Figura 12 – Tela de gravação de produto



Fonte: Criado pelo autor

Figura 13 – Tela de gravação de limpeza



Fonte: Criado pelo autor

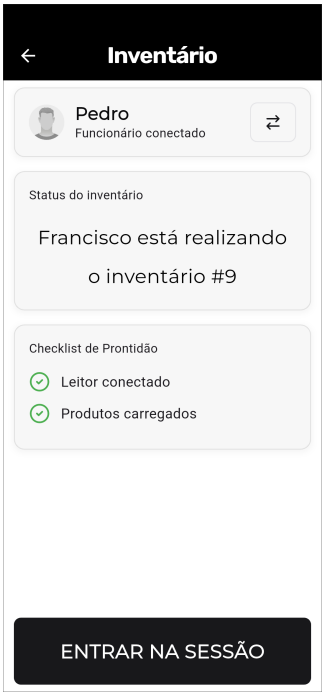
O segundo grande processo do aplicativo é o de inventário. Ao clicar no botão

de navegação "Inventário"na tela principal, o usuário irá para a tela de preparação de inventário. Nessa tela, o aplicativo realiza uma requisição para o servidor para tentar obter o inventário do dia. Se não houver um, ele cria e o atribui ao usuário logado na aplicação. Antes de entrar em um inventário, a interface mostra claramente qual das duas situações descritas anteriormente aconteceu (Figura 14 e Figura 15).

Figura 14 – Tela de *setup* de inventário ao criar um novo

Fonte: Criado pelo autor

Figura 15 – Tela de *setup* de inventário ao adentrar um existente



Fonte: Criado pelo autor

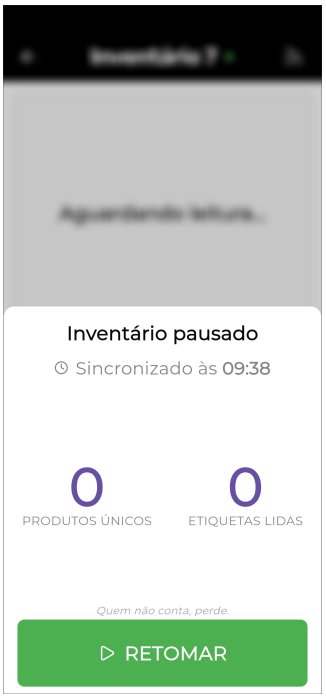
A tela de inventário (Figura 16) tem duas seções principais, um componente superior que mostra a contabilização do produto que está sendo lido atualmente, e um componente inferior com o histórico das outras leituras feitas naquele inventário. Caso seja necessário, é possível pausar o inventário (Figura 17) apertando no botão secundário que fica na parte mais baixa da tela.

A sincronização com o servidor das leituras feitas em um inventário é realizada periodicamente pelo app em plano de fundo, mas também pode ser feita manualmente por meio do botão de ação que fica no canto direito da barra superior do aplicativo (Figura 18). No último fluxo alternativo de inventário, se na tela de preparação de inventário, a conexão com a internet ou com o servidor backend não estiver disponível, é possível fazer a realização de um inventário totalmente local no próprio aplicativo (Figura 19). Nesse caso, não há sincronização automática do app, mesmo quando a conexão é restabelecida, ficando a cargo do usuário a sincronização manual dos dados.

Figura 16 – Tela de realização de inventário

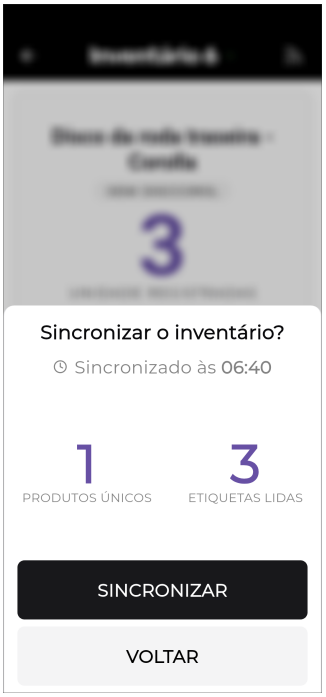
Fonte: Criado pelo autor

Figura 17 – Tela de pausa de inventário



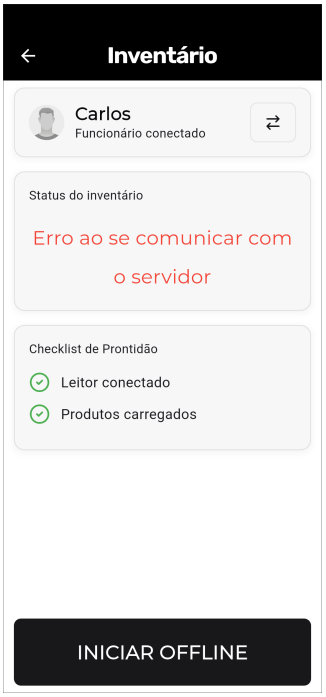
Fonte: Criado pelo autor

Figura 18 – Tela de sincronização de inventário



Fonte: Criado pelo autor

Figura 19 – Tela de *setup* de inventário offline

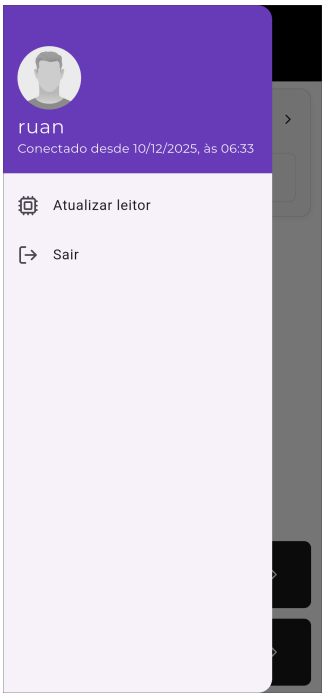


Fonte: Criado pelo autor

Por fim, o último fluxo do aplicativo é o de atualização de firmware. Essa ação

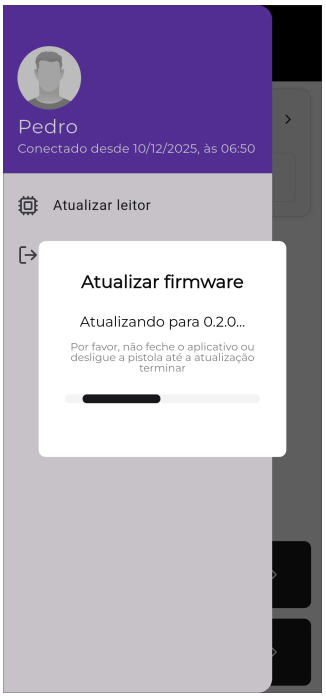
é disponível a partir de um componente deslizável localizado na parte esquerda da tela principal. Ao abri-lo (Figura 20), é possível verificar a sessão atual, encerrá-la, e também realizar a atualização do leitor. Ao clicar nessa opção, o firmware realiza uma busca no servidor pela versão mais atual disponível, e caso haja uma, ele realiza o download do arquivo binário correspondente e executa uma atualização *over-the-air* por meio da conexão Bluetooth com o leitor (Figura 21).

Figura 20 – Componente deslizável da tela principal



Fonte: Criado pelo autor

Figura 21 – Modal de atualização do leitor

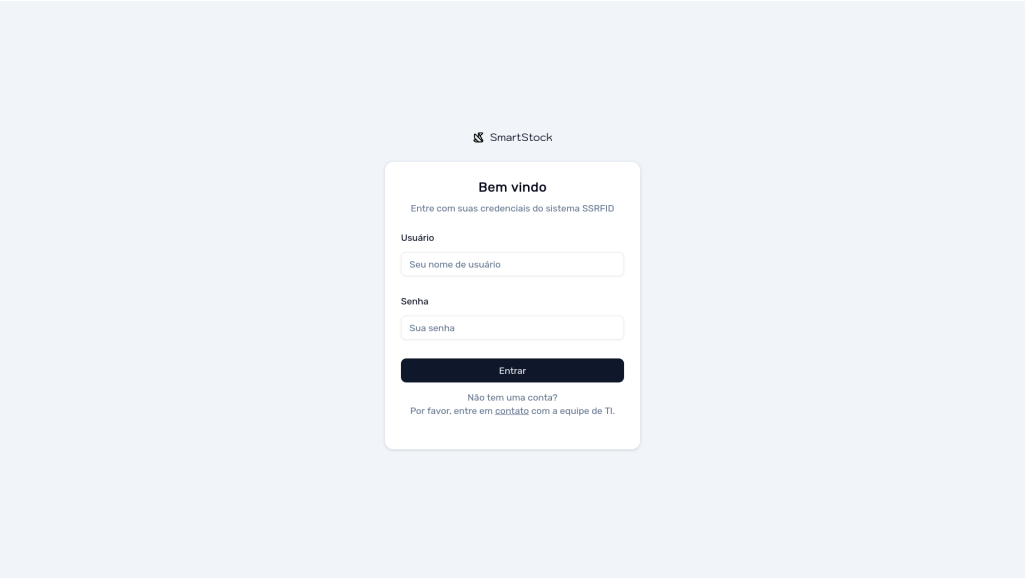


Fonte: Criado pelo autor

4.2 CLIENTE WEB

A primeira página da aplicação Web é a de login (Figura 22), onde o administrador deve entrar com as credenciais geradas para ele diretamente no backend ou por outro administrador, não sendo permitida a criação de conta por novos usuários no frontend por questões de segurança e controle administrativo.

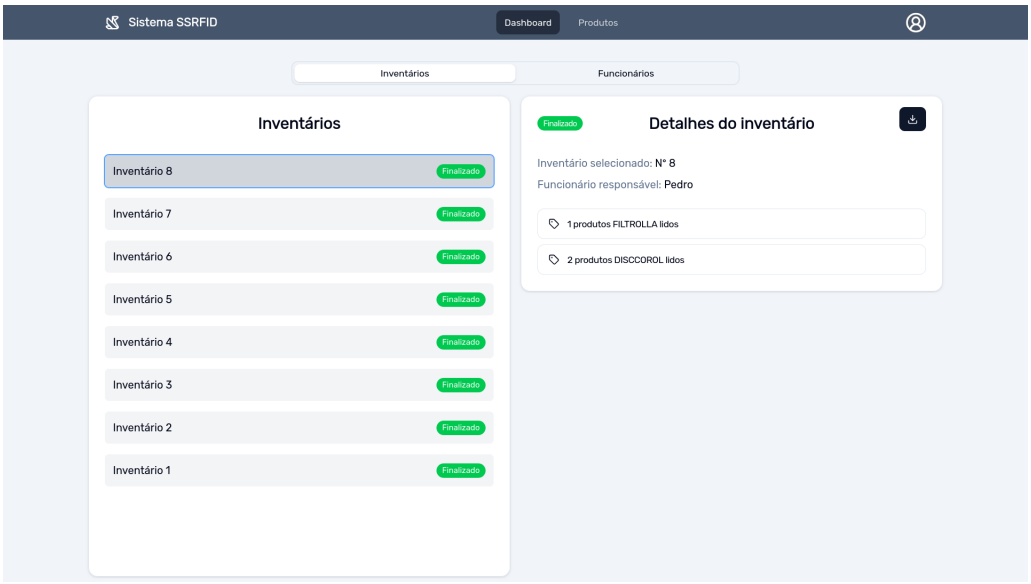
Figura 22 – Página de Login



Fonte: Criado pelo autor

Existem duas telas principais no sistema, uma para gerenciar os inventários, e outra para gerir os usuários. Na tela de gerenciamento de inventários (Figura 23), os inventários são listados no *card* esquerdo, enquanto o *card* direito mostra os detalhes de um inventário selecionado. É na seção de detalhes que é possível realizar o download do relatório de inventários fechados (como mostrado na Figura 24), e cancelar ou concluir inventários em aberto.

Figura 23 – Página de Inventários



Fonte: Criado pelo autor

Figura 24 – Relatório de inventário

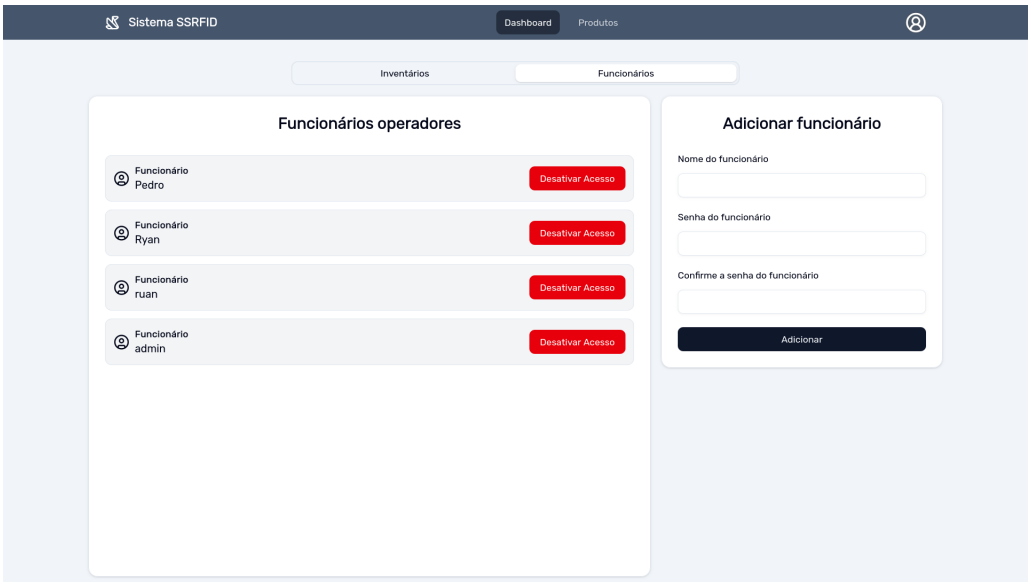
Relatório de Inventário RFID

ID	Funcionário	Status	Início	Fim	Duração (min)	Leituras	Peças
8	Pedro	FINALIZADA	10/12/2025 14:22	10/12/2025 14:33	10.95	2	3

Fonte: Criado pelo autor

A tela de gerência de usuários (Figura 25) segue uma interface semelhante à anterior, mas no lugar de um *card* de detalhes, na direita é exibido o formulário de cadastro de um novo usuário no sistema. A única interação com os usuários já cadastrados ocorre no canto direito dos itens listados, onde por meio de um botão é possível desativar e ativar o acesso de um usuário ao sistema.

Figura 25 – Página de Usuários



Fonte: Criado pelo autor

5 CONSIDERAÇÕES FINAIS

O produto final do desenvolvimento deste trabalho foi um MVP (Produto Mínimo Viável) dos clientes móvel e Web do sistema conjunto SmartStock. As soluções foram materializadas em uma plataforma Web desenvolvida em Next.js e um aplicativo móvel em Flutter, ambos integrados ao ecossistema SmartStock.

5.1 IMPLANTAÇÃO

No âmbito do programa EmbarcaTech, projeta-se a integração real da solução com empresas parceiras. Embora essa fase ocorra após a finalização deste relatório, a disponibilização do código-fonte sob licença MIT garante que a comunidade e empresas interessadas possam adotar ou expandir o sistema prontamente.

5.2 RESULTADOS DO DESENVOLVIMENTO

A comunicação via Bluetooth Low Energy entre o aplicativo móvel e o leitor RFID foi validada com o microcontrolador ESP32 do protótipo, provando ser ágil e uma forma de conexão válida. No campo Web, a troca de dados entre o cliente Web e o servidor backend é estruturada por meio de uma API REST e HTTPS, assegurando o tráfego seguro de dados. A arquitetura de ambos os clientes seguiu práticas modernas de código limpo, que garantem manutenibilidade e escalabilidade do software.

Ademais, este relatório também apresenta um modelo replicável de desenvolvimento de clientes para soluções de estoque no geral, visto que apesar do foco voltado para o uso em conjunto com a tecnologia RFID, é notória a facilidade de expansão dessas aplicações para contextos de estoque mais gerais. Com o desenvolvimento em ferramentas de código aberto, espera-se que a solução proposta seja um protótipo acessível e de valor para a comunidade e empresas de um modo geral.

5.3 TRABALHOS FUTUROS

O escopo da solução proposta aqui entrega a contagem de produtos, mas o usuário ainda precisa conferir manualmente se a contabilidade feita corresponde ao esperado. Como trabalhos futuros, para além de inventários, sugere-se a comunicação com um sistema de estoque tradicional, para que a solução tenha autonomia em conferir discrepâncias entre a contagem realizada e a esperada, automatizando ainda mais o processo de verificação de inventário.

REFERÊNCIAS

ALMEIDA, A. N. d.; SILVA, J. C. G. L. d.; ANGELO, H. Importância dos setores primário, secundário e terciário para o desenvolvimento sustentável. **Revista Brasileira de Gestão e Desenvolvimento Regional**, v. 9, n. 1, p. 111–125, 2013. Acessado em: 09 dez. 2025. Disponível em: https://www.rbgdr.net/revista/index.php/rbgdr/article/view/874.

Docker, Inc. **Docker: Accelerated Container Application Development**. 2025. Acessado em: 09 dez. 2025. Disponível em: https://www.docker.com/.

Figma, Inc. **Figma: The Collaborative Interface Design Tool**. 2025. Acessado em: 09 dez. 2025. Disponível em: https://www.figma.com/.

Fundação Instituto de Pesquisas Econômicas. **A importância do Terceiro Setor para o PIB no Brasil e em suas Regiões**. São Paulo, 2023. Acessado em: 09 dez. 2025. Disponível em: https://info.sitawi.net/terceiro-setor-pib-brasil.

IBRAHIM, N. *et al.* The effectiveness of web 2.0 tools training workshop using canva and figma in developing creative visual content. **Asian Journal of Assessment in Teaching and Learning**, v. 13, n. 2, p. 35–45, 2023. Acessado em: 09 dez. 2025. Disponível em: https://doi.org/10.37134/ajatel.vol13.2.4.2023.

JARTARGHAR, H. A. *et al.* React apps with server-side rendering: Next.js. **Journal of Telecommunication, Electronic and Computer Engineering**, v. 14, n. 4, p. 25–29, 2022. Acessado em: 09 dez. 2025. Disponível em: https://jtec.utem.edu.my/jtec/article/view/6192.

Lucide Contributors. **Lucide: Beautiful & consistent icons**. 2025. Acessado em: 09 dez. 2025. Disponível em: https://lucide.dev/.

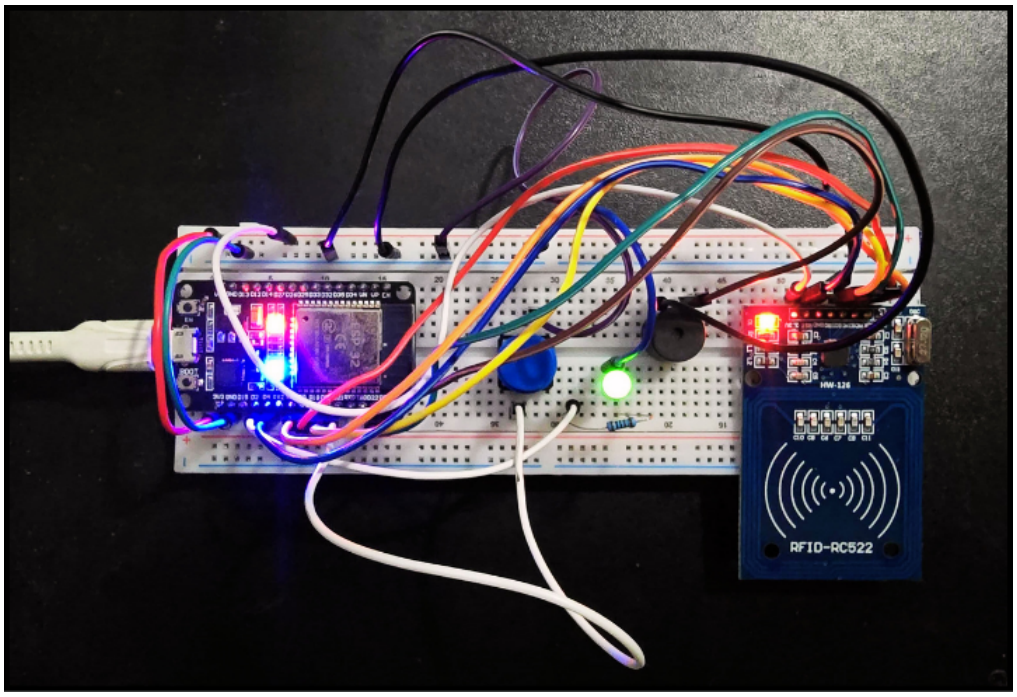
MARIMUTHU, K. *et al.* Android based college app using flutter dart. **Green Intelligent Systems and Applications**, v. 3, n. 2, p. 69–85, 2023. Acessado em: 09 dez. 2025. Disponível em: https://tecnoscientifica.com/journal/gisa/article/view/269.

ROKICKI, T.; MAURICE, C.; LAPERDRIX, P. Sok: In search of lost time: A review of javascript timers in browsers. In: **2021 IEEE European Symposium on Security and Privacy (EuroSP)**. [s.n.], 2021. p. 472–486. Acessado em: 09 dez. 2025. Disponível em: https://ieeexplore.ieee.org/document/9581218.

Softex. **EmbarcaTech: Programa de Residência Tecnológica em Sistemas Embarcados**. 2025. Acessado em: 09 dez. 2025. Disponível em: https://embarcatech.softex.br/.

YEE, M.-H.; GUHA, A. Do machine learning models produce typescript types that type check? **arXiv**, 2023. Acessado em: 09 dez. 2025. Disponível em: https://arxiv.org/abs/2302.12163.

APÊNDICE A – PROTÓTIPO DO LEITOR



Fonte: Criado pelo autor